CSEP 521 - Spring 2005
Assignment 6

Due 5/12/05

1. Suppose we are in the case where we know the probability of 0 is much more than 1/2, but we don't know it precisely. The question is how to design an adaptive Golomb code that will work well. One way to do this is by doubling the order until the first 1 is found, then use the optimal Golomb code from that point on. This is called the *doubling algorithm*. For example, suppose the input is $0^{12}10^{10}10^{13}1$. The following table show how the coding proceeds:

order	input	output	calculation for next order
1	0	1	$2 = 2 \times 1$
2	00	1	$4 = 2 \times 2$
4	0000	1	$8 = 2 \times 4$
8	000001	0101	$9 = \lceil -1/\log_2(12/13) \rceil$
9	0^9	1	$15 = \lceil -1/\log_2(21/22) \rceil$
15	01	00010	$8 = \lceil -1/\log_2(22/24) \rceil$
8	0^8	1	$11 = \lceil -1/\log_2(30/32) \rceil$
11	000001	01010	$9 = \lceil -1/\log_2(35/38) \rceil$

   Note that after coding $0^{12}1$ we have seen exactly one 1 out of 13 input symbols. This is why we switch to order 9. Similarly, after coding $0^{12}10^9$, we have seen exactly one 1 out of 22 total symbols so we switch to order 15, and so on.

   (a) Code the string $0^8 10^{15} 10^8 1$ using the doubling algorithm. What is the compression ratio?

   (b) Decode the string 111100110000101111 using the doubling algorithm. What is the compression ratio?

2. Let us try LZW on a special class of inputs too. Again assume the two symbol alphabet $\{0, 1\}$. Consider the following strategy for encoding the dictionary symbols from LZW. Start with a dictionary of size 2 and use just one bit to transmit a symbol. When the dictionary fills up we double its size to 4 and use two bits to transmit a word in the dictionary. This doubling happens when ever the dictionary fills.

   (a) Encode 0^6 and 0^{28} with this version of LZW.

   (b) Compute the length, as a function of n, of the encoding of 0^n with this version of LZW. (You may restrict yourself to n's of the form $n = k(k+1)/2$ if that helps.)

   (c) Encode 0^6 and 0^{28} using the γ-code to represent the dictionary symbols from LZW on the strings 0^6 and 0^{28}.

(d) Compute the length, as a function of n, of the encoding of 0^n using the γ-code to represent the dictionary symbols of LZW. (You may restrict yourself to n's of the form $n = k(k+1)/2$ if that helps.)

(e) How does LZW compare to adaptive Golomb and adaptive arithmetic coding for compressing binary strings with very few 1s? Explain the comparison in one brief paragraph.